Deep Multi-View Learning Using Neuron-Wise Correlation-Maximizing Regularizers

Kui Jia, Jiehong Lin, Mingkui Tan, and Dacheng Tao, *Fellow, IEEE*

***Abstract*— Many machine learning problems are concerned with discovering or associating common patterns in data of multiple views or modalities. Multi-view learning is one of the methods to achieve such goals. Recent methods propose deep multi-view networks via adaptation of generic deep neural networks (DNNs), which concatenate features of individual views at intermediate network layers (i.e., *fusion layers*). In this paper, we study the problem of multi-view learning in such end-to-end networks. We take a regularization approach via multi-view learning criteria, and propose a novel, effective, and efficient neuron-wise correlation-maximizing regularizer. We implement our proposed regularizers collectively as a *correlation-regularized network layer (CorrReg)*. CorrReg can be applied to either fully-connected or convolutional fusion layers, simply by replacing them with their CorrReg counterparts. By partitioning neurons of a hidden layer in generic DNNs into multiple subsets, we also consider a multi-view feature learning perspective of generic DNNs. Such a perspective enables us to study deep multi-view learning in the context of regularized network training, for which we present control experiments of benchmark image classification to show the efficacy of our proposed CorrReg. To investigate how CorrReg is useful for practical multi-view learning problems, we conduct experiments of RGB-D object/scene recognition and multi-view-based 3D object recognition, using networks with fusion layers that concatenate intermediate features of individual modalities or views for subsequent classification. Applying CorrReg to fusion layers of these networks consistently improves classification performance. In particular, we achieve the new state of the art on the benchmark RGB-D object and RGB-D scene datasets. We make the implementation of CorrReg publicly available.**

***Index Terms*— Multi-view learning, deep learning, regularization, normalization, canonical correlation analysis.**

I. Introduction

MANY machine learning problems concern with discovering or associating common patterns in data of multiple views or modalities. Typical applications include retrieving images from texts or vice versa, combining visual and audio signals for content understanding, and object recognition from visual observations of multiple modalities. Data of different views usually contain complementary information, whose statistical distributions in the high-dimensional measurements of individual views may also be different. Multi-view learning methods aim to exploit information contained in multiple views to better accomplish specified learning tasks. In this work, we take image classification, in particular multi-view or multi-modal object recognition (e.g., recognizing objects from RGB and depth images), as the primary example to study the problem of multi-view learning.

Given feature observations of different views, existing multi-view learning approaches learn latent space representations in either deterministic [1]–[4] or probabilistic manners [5]–[8]. The learning objective is to make resulting features of different views at each dimension of the latent space more *related* with each other, where relations may be measured by different metrics/criteria [4], [9], [10]. Among various techniques, Canonical Correlation Analysis (CCA) [1] and its extensions [2], [11], [12] are the most representative ones. For example, given two-view data, CCA learns pairs of linear projections so that in the projected space, features of both views are maximally correlated at corresponding dimensions.

Following the success of deep learning, deep multi-view learning methods [2], [13] are also proposed recently for learning deep features from multi-view data. These methods apply multi-view learning criteria (e.g., CCA) on top of multiple single-view deep networks (cf. Figure 1-(a) for an illustration); a two-stage scheme of iterative learning is usually adopted to train the network parameters, where view-specific features are learned until the very top layers, to which either a sequential step of multi-view criteria followed by the objectives of the specified learning tasks or regularized learning objectives that seek a balance between multi-view criteria and the final tasks of interest, are applied. Alternatively, one may design deep architectures that concatenate at intermediate network layers (*fusion layers*) output features of lower, parallel layer streams for individual views [3], [8], [14], followed by upper network layers for specified learning tasks (e.g., image classification, cf. Figure 1-(b) for an illustration). Such end-to-end networks have the advantage that the final tasks of interest are achieved directly at the network outputs. However, output features of the lower, parallel streams in such networks capture view-specific patterns, which may not be aligned in a common space for a ready fusion in the subsequent layers.

Manuscript received November 6, 2018; revised February 23, 2019; accepted April 3, 2019. Date of publication May 7, 2019; date of current version August 14, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61771201 and Grant 61602185, in part by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2017ZT07X183, and in part by the Guangdong Provincial Scientific and Technological Funds under Grant 2018B010107001. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Tao Mei. *(Corresponding author: Kui Jia.)*

K. Jia and J. Lin are with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: kuijia@scut.edu.cn; lin.jiehong@mail.scut.edu.cn).

M. Tan is with the School of Software Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: mingkuitan@scut.edu.cn).

D. Tao is with the School of Information Technologies, The University of Sydney, NSW 2006, Australia (e-mail: dacheng.tao@sydney.edu.au).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TIP.2019.2912356

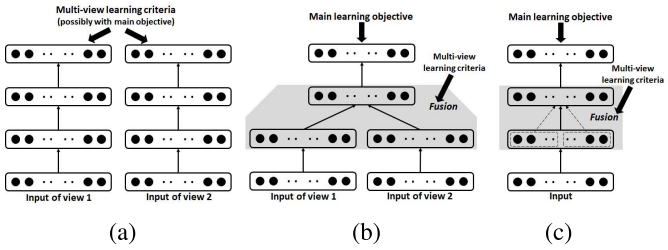

Fig. 1. Two-view illustrations of deep networks for multi-view learning, where *fusion layers* are inside shaded regions that concatenate features of individual views. (a) Deep multi-view features are learned by applying multi-view learning criteria, possibly together with the main learning objective, on top of multiple single-view deep networks; (b) a deep network takes as inputs data of multiple views/modalities for a specified learning task (e.g., multi-modal image classification), where features of individual views are concatenated at an intermediate layer (i.e., the fusion layer), and multi-view learning criteria can be imposed as a regularization on the fusion layer; (c) in generic DNNs, input neurons of a hidden layer can be partitioned into multiple subsets (represented as black circles grouped in different dashed boxes), and features learned at different subsets could be considered as multi-view features of the same input data.

To enjoy the advantage of end-to-end learning while collaboratively benefiting from different views, multi-view learning criteria could be exploited to improve the relations between resulting features of the lower, parallel streams. Under the framework of regularized function learning, this amounts to training network parameters by penalizing objectives of the main learning tasks with correlation-maximizing regularization at fusion layers (cf. Figure 1-(b)). In this work, we are interested in CCA criteria since they have long been the main workhorse for multi-view learning [10], [11], [15]. Empirical results also show that CCA based approaches outperform alternative ones in the context of deep multi-view representation learning [13]. Directly using CCA as the regularizer makes network training very expensive, which is also incompatible with the mini-batch based stochastic gradient descent (SGD), the commonly used algorithm in deep network training (cf. Section II for a discussion). Inspired by batch normalization [16], we propose in this paper a novel *neuron-wise correlation-maximizing regularizer*, and implement the proposed regularizers collectively as a *correlation-regularized network layer (CorrReg)*. CorrReg can be applied to either fully-connected (FC) or convolutional (conv) fusion layers, simply by replacing these layers with their CorrReg versions (cf. Figure 3 for an illustration). CorrReg fusion has the same computational complexity as the plain one does, which is significantly lower than that of CCA regularization.

We note that the fusion layer in a multi-view network of Figure 1-(b) is *computationally* equivalent to any hidden layer in a generic deep neural network (DNN), by partitioning neurons of the hidden layer into multiple subsets (cf. Figure 1-(c) for an illustration). This equivalence suggests a multi-view feature learning perspective of generic DNNs: when considering hidden neurons of generic DNNs as pattern detectors that characterize different patterns of the input data [17], features learned at each neuron subset of a hidden layer could be considered as a specific *view* of the input data. Ideally, each of such feature views is to learn salient or discriminative patterns of the input data, which should also be generalizable to unseen data. However, there exists an issue of *overfitting*, a phenomenon that specific subsets of layer neurons are trained to be co-adapted to certain patterns in the training data, but cannot generalize well on the held-out test data [18]. As suggested by traditional learning theory [19], the risk of overfitting could be severe for generic DNNs, since modern DNNs have large model capacities and are usually over-parameterized in the sense that they can be trained to fit randomly labeled datasets [20]. Such a risk for generic DNNs can be addressed implicitly by SGD training [21], [22], and explicitly by additional regularization [16], [18]. Our proposed CorReg takes the second regularization approach: improving correlations of neuron subsets reduces co-adaptation to possibly noisy or irrelevant, subset-specific patterns, and thus alleviates the problem of overfitting. Such a connection with generic DNNs enables us to study CorrReg in the context of regularized network training, and to compare with modern regularization techniques (e.g., Dropout [18]) for deep multi-view representation learning.

We finally summarize our contributions as follows.

- We study in this work the problem of learning deep representations from multi-view data in end-to-end networks. We take a regularization approach via multi-view learning criteria, and propose a novel, effective, and efficient neuron-wise correlation-maximizing regularizer. We implement our proposed regularizers collectively as a correlation-regularized network layer (CorrReg), which will be made publicly available. CorrReg can be applied to either FC or conv based fusion layers, simply by replacing these layers with their CorrReg versions. CorrReg fusion layer has the same computational complexity as a plain one does, which is significantly lower than that of CCA regularization.
- We consider a multi-view feature learning perspective of generic DNNs, by partitioning neurons of a hidden layer in a generic DNN into multiple subsets. Such a connection with generic DNNs enables us to study deep multi-view representation learning in the context of regularized network training, and to compare CorrReg with modern regularization techniques (e.g., Dropout [18]). We note that such a comparison is largely ignored in existing deep multi-view learning methods.
- To investigate the efficacy of CorrReg for regularization of network training, we conduct control experiments on benchmark image classification [23] using generic DNNs [24]–[27]. CorrReg consistently improves performance of these networks. To investigate how CorrReg is useful for practical multi-view learning problems, we conduct experiments of RGB-D object/scene recognition and multi-view based 3D object recognition, using networks with fusion layers that concatenate intermediate features of individual modalities or views for subsequent classification. Applying CorrReg to fusion layers of these networks consistently improves classification performance. In particular, we achieve the new state of the art on the benchmark RGB-D object [28] and RGB-D scene [29] datasets.

II. The Proposed Neuron-Wise Correlation-Maximizing Regularizers

In this section, we first use generic DNNs to present our proposed regularization method, and explain how our method is applied to their hidden layers. Our method is readily applied to fusion layers of deep networks that take practical multi-view data, which will be introduced in Section III.

We start with a DNN composed of L FC layers. Denote its network parameters as $\Theta = \{\boldsymbol{W}^l, \boldsymbol{b}^l\}_{l=1}^L$, where $\boldsymbol{W}^l$ and $\boldsymbol{b}^l$ are respectively the weight matrix and bias vector associated with the l^{th} network layer. In the setting of supervised learning, given M training samples $\mathcal{S} = \{\boldsymbol{s}^i\}_{i=1}^M$ of categorical data, the network parameters in Θ are optimized by minimizing the empirical risk $\frac{1}{M}\sum_{i=1}^M Loss(\boldsymbol{s}^i; \Theta)$, where $Loss(\cdot)$ is a properly chosen loss function, e.g., cross-entropy loss for image classification, and optimization is typically based on SGD or its variants [30]. As discussed in Section I, DNNs of high model capacities are able to learn complex functions but susceptible to overfitting. To remedy, one may apply regularization to reduce their model capacities. Adding regularization to the network training objective results in the following optimization problem

$$\min_{\Theta} \frac{1}{M}\sum_{i=1}^{M} Loss(\boldsymbol{s}^i; \Theta) + \lambda R(\Theta), \tag{1}$$

where $R(\cdot)$ is the regularizer to be specified, and λ is a trade-off parameter.

In this work, we are interested in regularizing network training using CCA based multi-view learning criteria [2], [13]. More specifically, for a specified l^{th} network layer and a training sample $\boldsymbol{s}$, denote as $\boldsymbol{x} \in \mathbb{R}^{n_l}$ the feature vector of $\boldsymbol{s}$ computed all the way up from the input layer to the l^{th} layer. The l^{th} layer computes $f(\boldsymbol{z}) = f(\boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}) \in \mathbb{R}^{n_{l+1}}$, where $f(\cdot)$ is an element-wise nonlinear activation function such as ReLU [31], $\boldsymbol{W} \in \mathbb{R}^{n_l \times n_{l+1}}$ and $\boldsymbol{b} \in \mathbb{R}^{n_{l+1}}$ are the weight matrix and bias vector respectively, and where we omit the superscript l to make the following notations of better clarity. By randomly partitioning n_l input neurons/dimensions of the l^{th} layer into two subsets [1], we get feature subvectors $\boldsymbol{x}_1 \in \mathbb{R}^{n_{l_1}}$ and $\boldsymbol{x}_2 \in \mathbb{R}^{n_{l_2}}$ from $\boldsymbol{x}$, and the corresponding weight submatrices $\boldsymbol{W}_1 \in \mathbb{R}^{n_{l_1} \times n_{l+1}}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{n_{l_2} \times n_{l+1}}$ from $\boldsymbol{W}$. We simply have $\boldsymbol{W}^\top \boldsymbol{x} = \boldsymbol{W}_1^\top \boldsymbol{x}_1 + \boldsymbol{W}_2^\top \boldsymbol{x}_2$. Discussions in Section I suggest that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ can be analogously considered as two feature views of the input data. The co-adaptation of features in each view is likely to cause network training to pay more attention to the extraction of view-specific patterns, rather than the category related patterns that are desired to be learned by network training. To address this issue and benefit more from both views of the features, one may use CCA criteria to regularize network training, which aim to increase the feature correlations between the two views. Given the two-view features $\boldsymbol{X}_1 = [\boldsymbol{x}_1^1, \ldots, \boldsymbol{x}_1^M]$ and $\boldsymbol{X}_2 = [\boldsymbol{x}_2^1, \ldots, \boldsymbol{x}_2^M]$ of training set $\mathcal{S}$, applying CCA to the l^{th} network layer amounts to optimizing $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ by

$$\begin{aligned} &\max_{\boldsymbol{W}_1, \boldsymbol{W}_2} \frac{1}{M}\mathrm{tr}\left(\boldsymbol{W}_1^\top \boldsymbol{X}_1 \boldsymbol{X}_2^\top \boldsymbol{W}_2\right) \\ &\text{s.t. } \frac{1}{M}\boldsymbol{W}_1^\top \boldsymbol{X}_1 \boldsymbol{X}_1^\top \boldsymbol{W}_1 = \frac{1}{M}\boldsymbol{W}_2^\top \boldsymbol{X}_2 \boldsymbol{X}_2^\top \boldsymbol{W}_2 = \boldsymbol{I}, \end{aligned} \tag{2}$$

where $\boldsymbol{I}$ is an identity matrix of compatible size. The data matrices have been assumed centered for simplicity. Applying the above problem to the l^{th} network layer also assumes implicitly that $n_{l+1} \leq \min(n_{l_1}, n_{l_2})$.

When directly using (2) as the regularizer $R(\cdot)$ in (1), network training requires solving (2) with stochastic optimization methods. Unfortunately, as pointed out in [9], the objective (2) does not easily admit a stochastic optimization due to the involvement of data covariance matrices in the constraints. One may use batch gradient descent to solve (2). It computes gradients of the correlation objective w.r.t. the CCA projected features $\boldsymbol{W}_1^\top \boldsymbol{X}_1$ and $\boldsymbol{W}_2^\top \boldsymbol{X}_2$, which in turn will be used through back-propagation to compute the gradients w.r.t. $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, and w.r.t. all the network parameters in the layers below [2]. This is expensive as it involves computation of covariance matrices (of $\boldsymbol{W}_1^\top \boldsymbol{X}_1$ and $\boldsymbol{W}_2^\top \boldsymbol{X}_2$), their inverse square roots, and also performing matrix singular value decomposition (SVD). One may nevertheless try mini-batch based gradient descent to solve (2), which, however, may produce singular data covariance matrices; [2] also points out that solving (2) by mini-batch based stochastic optimization empirically gives unsatisfactory results. Given these challenges of directly using CCA as the regularizer $R(\cdot)$, we are motivated to find an alternative way to improve the correlations between the two feature views $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$.

Inspired by batch normalization [16], we propose to simplify the full CCA regularization in DNNs by considering the following two aspects. Firstly, instead of learning $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ to increase correlations at all the n_{l+1} dimensions of the resulting features jointly, we propose to learn $\boldsymbol{w}_{1,i}$ and $\boldsymbol{w}_{2,i}$, $i \in \{1, \ldots, n_{l+1}\}$, independently for each output neuron of the l^{th} layer, where $\boldsymbol{w}_{1,i}$ and $\boldsymbol{w}_{2,i}$ are the i^{th} columns of $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ respectively. Note that such a decoupling suggests that the resulting features at the n_{l+1} dimensions could be correlated, but at the same time the constraint of $n_{l+1} \leq \min(n_{l_1}, n_{l_2})$ is also relaxed, enabling its flexible use in DNNs as tasks demand. Secondly, we use mini-batches of m training samples, rather than all the M ones, to approximate the statistics (i.e., mean and variance) necessary for computing correlations. This second simplification is enabled by the independent neuron-wise learning of $\boldsymbol{w}_{1,i}$ and $\boldsymbol{w}_{2,i}$: since in the joint case, the size m of mini-batches is required to be big enough to avoid singularity of covariance matrices.

For a specified output neuron of the l^{th} layer, we first introduce the (scalar) random variables Y_1 and Y_2 whose samples are respectively computed as $y_1 = \boldsymbol{w}_1^\top \boldsymbol{x}_1$ and $y_2 = \boldsymbol{w}_2^\top \boldsymbol{x}_2$, as illustrated in Figure 2, where we omit the neuron index for notational clarity. Given a mini-batch of m training samples, we propose in this work the following *neuron-wise*

[1]For simplicity, we only consider in this work the case of partitioning neurons of a hidden layer in a generic DNN into two subsets.

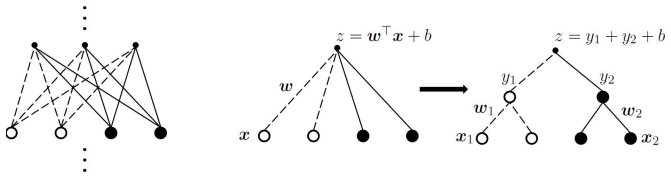

Fig. 2. Two sub-layers are formed by randomly partitioning the input neurons and the corresponding weights of a network layer into two subsets (Left). For a specified output neuron of the layer, two internal features $y_1 = \boldsymbol{w}_1^\top \boldsymbol{x}_1$ and $y_2 = \boldsymbol{w}_2^\top \boldsymbol{x}_2$ can be computed from the partition (Right). Blank circles and dashed lines represent one sub-layer, and filled circles and solid lines represent the other.

correlation-maximizing regularizer

$$Corr(Y_1, Y_2) \approx \frac{\sum_{i=1}^m (y_1^i - \mu_1)(y_2^i - \mu_2)}{\sqrt{\sigma_1^2 \sigma_2^2 + \epsilon}}, \tag{3}$$

where the scalar ϵ is introduced for numerical stability, and

$$\begin{aligned} y_1^i &= \boldsymbol{w}_1^\top \boldsymbol{x}_1^i \quad i = 1, \ldots, m, \\ y_2^i &= \boldsymbol{w}_2^\top \boldsymbol{x}_2^i \quad i = 1, \ldots, m, \\ \mu_1 &= \frac{1}{m} \sum\nolimits_{i=1}^m y_1^i, \quad \mu_2 = \frac{1}{m} \sum\nolimits_{i=1}^m y_2^i, \\ \sigma_1^2 &= \sum\nolimits_{i=1}^m (y_1^i - \mu_1)^2, \quad \sigma_2^2 = \sum\nolimits_{i=1}^m (y_2^i - \mu_2)^2. \end{aligned}$$

Based on the neuron-wise regularizer (3), we specify the general objective function (1) as the following regularized problem to improve network training

$$\min_{\Theta} \frac{1}{M} \sum_{i=1}^M Loss(\boldsymbol{s}^i; \Theta) - \lambda \sum_{g \in \mathcal{G}} Corr(\mathcal{S}; \theta_g), \tag{4}$$

where g indexes the group $\mathcal{G}$ of neurons in the network layers that are specified to apply regularization, θ_g denotes a subset of network parameters Θ that are involved in the computation of the incoming features of the neuron g, and we have slightly abused the use of notation $Corr$ with that in (3). Note that θ_g and $\theta_{g'}$ may contain overlapped parameters, i.e., those parameters associated with the common layers below g and g'. The main objective (4) can be optimized using SGD or its variants [30], by sampling mini-batches of training samples in iterative steps. Details are presented shortly. Complexity analysis presented in Section II-A.1 shows that compared with standard SGD training, our regularizer (3) increases computation cost only by a constant factor.

A. Correlation-Regularized Network Layer

We still use the illustration in Figure 2 as the running example. In the forward pass of a mini-batch of size m, any output neuron of the l^{th} layer that is specified to apply the regularization (3) computes $y_1^i = \boldsymbol{w}_1^\top \boldsymbol{x}_1^i$ and $y_2^i = \boldsymbol{w}_2^\top \boldsymbol{x}_2^i$, $i = 1, \ldots, m$, which give output features of the neuron, before nonlinear activation $f(\cdot)$, as $z^i = y_1^i + y_2^i + b$, $i = 1, \ldots, m$, where b is the bias associated with this neuron. [2] In the backward pass, we need to compute the gradients of the neuron-wise regularizer w.r.t. the weight vectors $\frac{\partial Corr}{\partial \boldsymbol{w}_1}$ and $\frac{\partial Corr}{\partial \boldsymbol{w}_2}$, and also those w.r.t. the input features $\frac{\partial Corr}{\partial \boldsymbol{x}_1^i}$, $\frac{\partial Corr}{\partial \boldsymbol{x}_2^i}$, $i = 1, \ldots, m$. These gradients can be derived via multi-variable chain rule, and we give their explicit forms in supplementary material. Note that the later ones are used to compute through back-propagation gradients of the regularizer w.r.t. network parameters in the lower layers that are also involved in the computation of $\{z^i\}_{i=1}^m$.

We usually apply (3) to all the n_{l+1} output neurons of the specified l^{th} layer. To make regularized network training efficient, we note that for these n_{l+1} neurons, their respective internal features and statistics, i.e., $\{y_1^i\}_{i=1}^m$, $\{y_2^i\}_{i=1}^m$, μ_1, μ_2, σ_1^2, and σ_2^2, and also the corresponding gradients can be computed independently and in parallel. Given the mini-batch of layer inputs of the two-view features $\boldsymbol{x}^i = [\boldsymbol{x}_1^i, \boldsymbol{x}_2^i]$, $i = 1, \ldots, m$, we write gradients of the n_{l+1} neuron-wise regularizers in the compact forms as $\left[\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_1^i}, \frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_2^i}\right]$, $i = 1, \ldots, m$, and $\left[\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_1}, \frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_2}\right]$, where $\boldsymbol{Corr}$ denotes the n_{l+1} correlation objectives compactly. More specifically, $\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_1^i}$ *sums* the gradients of neuron-wise regularizers in the layer w.r.t. the input $\boldsymbol{x}_1^i$, and $\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_1}$ *independently computes* the gradient of each neuron-wise regularizer w.r.t. its associated weight vector $\boldsymbol{w}_1$; the same operations apply to $\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_2^i}$ and $\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_2}$.

In other words, we implement our proposed scheme (3) as *a correlation-regularized network layer* (CorrReg): in the forward pass, the computation is the same as a standard network layer, i.e., we do not explicitly compute the internal features $\boldsymbol{y}_1 = \boldsymbol{W}_1^\top \boldsymbol{x}_1$ and $\boldsymbol{y}_2 = \boldsymbol{W}_2^\top \boldsymbol{x}_2$, and instead we directly compute $\boldsymbol{z} = \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}$, followed by element-wise nonlinear activation; in the backward pass, we compute the gradients of the correlation-regularized loss w.r.t. layer weights and layer inputs, which we spell out as

$$\begin{aligned} &\frac{1}{m} \sum\nolimits_{i=1}^m \frac{\partial Loss(\boldsymbol{x}^i)}{\partial \boldsymbol{W}} - \lambda \left[\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_1}, \frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{W}_2}\right] \\ &\frac{\partial Loss(\boldsymbol{x}^i)}{\partial \boldsymbol{x}^i} - \lambda \left[\frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_1^i}, \frac{\partial \boldsymbol{Corr}}{\partial \boldsymbol{x}_2^i}\right] \quad i = 1, \ldots, m, \end{aligned}$$

where $\frac{\partial Loss(\boldsymbol{x}^i)}{\partial \boldsymbol{W}}$ and $\frac{\partial Loss(\boldsymbol{x}^i)}{\partial \boldsymbol{x}^i}$ can be computed via standard back-propagation. The gradient of the main loss w.r.t. the layer bias vector $\boldsymbol{b}$ can be obtained in the same way.

1) Analysis of Computational Complexity: We analyze the additional computation cost incurred by imposing CorrReg on a network layer. Consider an l^{th} layer that computes $f(\boldsymbol{z}) = f(\boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}) \in \mathbb{R}^{n_{l+1}}$ for a mini-batch of m samples, and that has layer parameters $\boldsymbol{W} \in \mathbb{R}^{n_l \times n_{l+1}}$ and $\boldsymbol{b} \in \mathbb{R}^{n_{l+1}}$. Assume arithmetic with individual elements has complexity $\mathcal{O}(1)$. In the forward pass, the computation for the layer with or without CorrReg is the same, and its complexity is $\mathcal{O}(mn_l n_{l+1})$. In the backward pass, without using CorrReg the complexity for back-propagating gradients through this layer is $\mathcal{O}(mn_l n_{l+1})$. By simplifying the gradient formulas given in supplementary material and also writing them in

[2] For each neuron that is applied the regularization, we have ever introduced trainable scalar parameters v_1 and v_2 to re-scale the internal features y_1 and y_2, i.e., $z = v_1 y_1 + v_2 y_2 + b$, which is similar to the scheme introduced in [32]. In our experiments this alternative scheme does not necessarily improve the performance.

matrix forms, we have the same complexity of $\mathcal{O}(mn_l n_{l+1})$ when using CorrReg. In summary, the complexity by imposing CorrReg on a network layer increases only by a constant factor. In contrast, using the CCA objective (2) as the regularizer involves computing inverse square root of covariance matrices of the size $n_{l+1} \times n_{l+1}$, and also performing SVD for matrix of the same size (one may refer to [2] for gradient formulas of CCA objective); it has the overall complexity of $\mathcal{O}(mn_l n_{l+1} + mn_{l+1}^2 + n_{l+1}^3)$ in the backward pass, which is significantly worse than that of CorrReg.

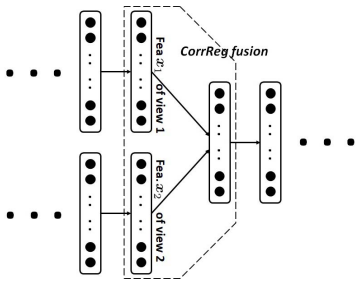

Fig. 3. Illustration of a *CorrReg fusion* layer (inside the dashed box).

B. Correlation-Regularized Convolutional Networks

Our proposed CorrReg can regularize both FC and conv layers. As discussed above, for an FC layer computing $f(\boldsymbol{z}) = f(\boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b})$, we apply regularization to the input $\boldsymbol{z}$ of f by performing a random two-way partition on feature dimensions of $\boldsymbol{x}$, producing two internal features $\boldsymbol{y}_1 = \boldsymbol{W}_1^\top \boldsymbol{x}_1$ and $\boldsymbol{y}_2 = \boldsymbol{W}_2^\top \boldsymbol{x}_2$, where the partition is fixed once determined. CorrReg is indeed to improve correlations between each dimension pair of $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$. It is straightforward to extend the above scheme of *single* two-way partition in CorrReg to its version of *multiple* two-way partitions. For a specified FC layer, one may simply perform multiple, random two-way partitions on feature dimensions of $\boldsymbol{x}$; each of them produces their respective internal features, and also their respective gradients w.r.t. layer weights and layer inputs. The overall regularization imposed on this layer can be obtained by averaging the gradients from these multiple two-way partitions. Experiments investigating the efficacy of this scheme of multiple two-way partitions are reported in Section V-A.

For a conv layer, we perform single or multiple random two-way partitions on its input feature maps in the same way as for an FC layer. Each two-way partition produces two internal feature maps (corresponding to y_1 and y_2 in Figure 2) for each output feature map of the layer. Although features/observations at nearby locations of an image are generally correlated, we do not explicitly exploit such correlations. Instead, we independently apply regularization at each spatial location/pixel of each output feature map of the layer, so that correlations between the corresponding spatial locations in each pair of the internal feature maps are improved, where regularization is again applied before nonlinear activation. Applying our proposed CorrReg to modern architectures of DNNs (e.g., ConvNets [33], variants of ResNets [25], [26], or DenseNets [27]) is very simple: one simply replaces FC or conv layers with their CorrReg versions.

III. Use of CorrReg Fusion in Deep Representation Learning from Multi-View Data

We present in this section how CorrReg can be readily applied to deep networks that take as inputs data of multiple views/modalities and learn deep representations from them. Different from generic DNNs, these networks by design have lower, parallel streams for data of individual views, and it is natural to apply CorrReg to the *fusion* layers where features of different views are concatenated for use in the subsequent layers. The fusion layer in such a network is usually based on FC or conv layers. To use CorrReg, one may simply treat output features of two parallel streams as the input two-view features of CorrReg (corresponding to $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in Section II), and replace the fusion layer with its CorrReg version, resulting in a *CorrReg fusion* layer. Figure 3 gives the illustration. In the following, we take network architectures used in our RGB-D recognition experiments to instantiate the use of CorrReg fusion.

Our networks for RGB-D object recognition and scene recognition are based on ConvNets [34] and ResNets [25]. Take an 8-layer ConvNet as the example. We modify it by enabling it to take inputs of both RGB and depth channels. The 8-layer ConvNet is composed of two lower, parallel streams followed by an upper, single stream. Outputs of the two lower streams are concatenated as inputs of the upper stream, and CorrReg is applied to the first layer of the upper stream, which thus becomes a CorrReg fusion layer. In this work, we also investigate the effects when the CorrReg fusion layer is at different "heights" (lower, middle, or upper layers) of the network. Figure 5 gives the illustration. Adaptation of ResNets is similar as above. We use such adapted network architectures for experiments of RGB-D object recognition and scene recognition in Section V-B.

IV. Relations with Existing Works

Our proposed CorrReg method is related to four categories of existing research: regularization of generic DNNs, deep learning research that specially focuses on multi-view representation learning, RGB-D object/scene recognition, and multi-view recognition of 3D object shapes. We respectively discuss the relations as follows.

A. Network Regularization

In the literature of DNNs, various regularization techniques have been proposed to address the issue of overfitting in network training, including the traditional early stopping, weight decay, and data augmentation, and also the more recent Dropout [18], Dropconnect [35], batch normalization [16], and all conv layer based networks [36], [37]. Among them, Dropout and Dropconnect are most related to our proposed method. In the original proposal of Dropout [18], each hidden neuron is randomly dropped (usually with a probability of 0.5) at each training iteration, and the network is then updated on weights that are connected to the remaining neurons. During inference, all network weights are used after halving their values. Baldi and Sadowski [38] quantitatively analyze that

random operations of dropout training and the associated inference can be understood as a good approximation to the expectation of outputs of a subnetwork ensemble, by introducing a bridging quantity Normalized Weighted Geometric Mean. They further show that the expectation of dropout gradients w.r.t. a network weight is approximately the gradient of the subnetwork ensemble regularized by *adaptive weight decay*. Wager *et al.* [39] present alternative interpretation of dropout training as adaptive weight decay by treating dropout as feature noising in generalized linear models. Analysis similar to [38] can be applied to Dropconnect [35] by randomly dropping weight connections rather than network neurons.

Dropout and Dropconnect achieve regularization by first sampling features/subnetworks (of shared weights), and then averaging over outputs of the subnetwork ensemble. Different from them, our CorrReg scheme explicitly increases the correlations between (internal) features of different views, and regularization is achieved by suppressing view-specific noisy patterns. We empirically show in this work the usefulness of CorrReg in improving network training, and leave its theoretical connections with classical regularization to future research. Note that a few recent methods of network regularization explicitly reduce correlations across dimensions of the output features of a layer [40], or achieve similar effects by enforcing orthogonality of the layer weight matrix [41], [42]. These methods impose regularization complementary to our proposed CorrReg, and we are interested in the investigation of their combined use in future research.

B. Deep Multi-View Representation Learning

Recent deep multi-view representation learning methods include those based on CCA [2], [13], [43] and those based on auto-encoders (AE) [3]. AE based methods typically learn a shared bottleneck layer on top of lower view-specific layers, and the learned joint representation at the bottleneck layer is used for reconstruction of multiple views. Deep CCA [2] directly applies CCA to the output layers of two deep networks, so that the learned networks can produce maximally correlated features at the output layers. Wang *et al.* [13] extend deep CCA as Deep Canonically Correlated Auto-Encoders, by balancing the correlation objective between the two views with their respective reconstruction objectives.

Most of existing deep multi-view learning works take a two-stage strategy for the final tasks of interest: they first learn from data of multiple views/modalities deep features in a common space, and then use the learned features in the common space either to train classifiers for multi-view or across-view classification, or to reconstruct data of missing views. In contrast, our use of correlation based multi-view learning is to regularize training of end-to-end networks, where parameters that project multi-view data into a common space are exactly those of an intermediate network layer.

C. RGB-D Object and Scene Recognition

RGB-D object recognition [8], [44]–[46] has drawn research attention recently as a typical application of multi-view learning. Lai *et al.* [28] collect the first large-scale, hierarchial RGB-D object dataset using a Kinect camera; they show that depth information substantially helps object recognition, by concatenating hand-crafted depth features with those of RGB ones, and using the concatenated features for classification. In [14], a hierarchical learning model of Convolutional and Recursive Neural Networks (CNNs and RNNs) are proposed, where CNNs are used for learning low-level features and RNNs with random weights for efficiently extracting higher-order features; this combined model is applied to RGB and depth images separately, and the resulting features are concatenated for classification.

Eitel *et al.* [45] propose a multi-modal deep learning architecture for RGB-D object recognition, which fuses, via feature concatenation, outputs of two parallel streams of modality-specific subnetworks (composed of conv and FC layers), and uses two additional FC layers for feature transformation and softmax classification; the whole network is trained via standard back-propagation with no consideration of multi-view learning/regularization criteria. Built on top of two parallel streams of ResNets (after removing their respective last layers of classifier) [25], a Correlated and Individual Multi-modal (CIM) learning layer is proposed in [47]; CIM aims to learn, in a discriminative and complementary manner, both correlated and modality-specific features from output vectors of the two ResNets, where "correlation" is measured by the Euclidean distance between projected features of the two ResNets' outputs; parameters of the whole network in [47] are updated in an alternating manner: those of the two lower streams of ResNets are updated after updating of the CIM parameters (projection matrices) converges. In [48], a deep learning framework termed MDSI-CNN is proposed to learn highly discriminative and spatially invariant multi-modal feature representations at different hierarchical levels. The problem of RGB-D image classification with limited training samples is addressed in [49]. It takes a domain adaptation approach and enforces the prediction consistency between two classifiers that are respectively learned either from the combined RGB and depth features or from the RGB features alone. Results on RGB-D object recognition show the efficacy of the proposed approach.

Methods for RGB-D scene recognition [50]–[53] largely follow those of RGB-D object recognition. In particular, multi-modal deep architectures similar to that of [45] are still the main workhorse to get good recognition performance. We also use such a type of networks for RGB-D recognition, but with our proposed CorrReg fusion layers that have in-built neuron-wise correlation regularization. Training of CorrReg fusion based networks has no difference from standard back-propagation.

D. Multi-View Recognition of 3D Object Shapes

Recent research shows that multi-view images are of a promising representation for recognition of 3D object shapes. Given a 3D object model (mesh), multiple 2D images can be rendered by placing virtual cameras around the object, and recognition is based on the rendered 2D images of multiple views. Among recent methods, MVCNN [54] is a representative one that uses parallel streams of conv layers to extract

TABLE I

ERROR RATES (%) ON CIFAR-10 [23] WHEN APPLYING CORRREG, WITH VARYING NUMBERS nReg OF RANDOM TWO-WAY PARTITIONS, TO DIFFERENT LAYERS OF A VARIANT OF LENET [24]. SETTING nReg AS 0 INDICATES NO CORRREG IS APPLIED TO ANY NETWORK LAYER. EXPERIMENTS OF EACH SETTING ARE RUN FOR 5 TIMES, AND RESULTS ARE IN THE FORMAT OF MEAN (STANDARD DEVIATION)

	No CorrReg	CorrReg Conv2	CorrReg Conv3	CorrReg FC4	CorrReg FC5	CorrReg FC6
$\texttt{nReg} = 0$	17.42 (0.16)	-	-	-	-	-
$\texttt{nReg} = 1$	-	17.15 (0.11)	17.14 (0.25)	17.13 (0.27)	16.75 (0.20)	16.92 (0.30)
$\texttt{nReg} = 3$	-	17.17 (0.21)	17.19 (0.17)	17.25 (0.23)	16.81 (0.28)	17.08 (0.11)
$\texttt{nReg} = 5$	-	17.08 (0.13)	17.23 (0.23)	17.28 (0.15)	16.64 (0.20)	16.84 (0.18)
$\texttt{nReg} = 10$	-	17.15 (0.40)	17.12 (0.12)	17.16 (0.20)	16.87 (0.18)	17.14 (0.31)

features from individual views, and then aggregates these features as a global signature simply via max pooling across different views. Subsequent works improve over MVCNN by strengthening interaction among feature learning of individual views. For example, MHBN [55] uses harmonized bilinear pooling to aggregate local features, and GVCNN [56] proposes a group-view framework to model correlations among different views at a hierarchy of multiple levels. In this work, we adapt the architectural design of MVCNN by incorporating into it CorrReg fusion layers. We use such an adapted architecture to verify the usefulness of CorrReg for multi-view based 3D object recognition.

V. EXPERIMENTS

In this section, we first present control experiments of image classification to investigate the effectiveness of our proposed CorrReg for regularization of network training. We use generic DNNs including ConvNet (LeNet) [24], and modern deep architectures of ResNet [25], Wide ResNet [26], DenseNet [27], and ResNeXt [57]. These experiments are conducted on the benchmark datasets of CIFAR-10, CIFAR-100 [23], and ImageNet [58]. We then present experiments of RGB-D object/scene recognition and multi-view 3D object recognition to evaluate the usefulness of CorrReg for practical multi-view learning problems. We use the benchmark datasets of RGB-D object [28], RGB-D scene [29], and ModelNet40 [59] for these experiments, and compare with the state-of-the-art results.

We use cross-entropy loss to train all these networks. Training is based on SGD with momentum. Without mentioning otherwise, we use mini-batches of size 128, momentum of 0.9, and weight decay of 0.0001; network parameters are initialized using Gaussian random weights; batch normalization is applied, before ReLU nonlinearity, in all networks to accelerate their training. In each experiment, the initial learning rate, the value of λ in (4), and also the dropping rate of Dropout (when using Dropout regularization) are determined by using 10% of training samples as the validation set. As (4) suggests, we use constant λ values for all neurons that are specified to apply CorrReg. Learning rates are decayed at the rate of 0.1 when learning curves plateau. Our implementation and experiments are based on the Torch library [60].

A. Control Experiments of Image Classification

We use the CIFAR-10 dataset for our controlled studies on a plain ConvNet (a variant of LeNet [24]). The CIFAR-10 dataset consists of 10 object categories of $60,000$ color images of size 32×32 ($50,000$ training and $10,000$ testing ones). We follow [61] and preprocess the data using global contrast normalization and ZCA whitening. Our used LeNet variant consists of 3 conv layers, followed by 3 FC layers. Max or average pooling layers are applied after each conv layer. More layer specifics are given in supplementary material.

We first investigate the regularization effects when applying CorrReg to different network layers. To this end, we replace each of the network layers (except the input one), namely Conv2, Conv3, FC4, FC5, and FC6, with their CorrReg versions respectively, and compare the recognition performance. As indicated in Section II-B, the scheme of multiple (random) two-way partitions can be used when applying CorrReg to any network layer. We also investigate how different numbers nReg of two-way partitions in CorrReg achieve regularization, for which we set $\texttt{nReg} = 1, 3, 5$, or 10. Note that when $\texttt{nReg} = 1$, we use the first half neurons/feature maps of the layer as a subset, and use the other half as the second subset; when $\texttt{nReg} > 1$, regularization is achieved by averaging over those of the multiple two-way partitions. We run experiments of each setting (the layer/nReg pair) for 5 times, and report results in the format of mean (standard deviation). Error rates reported in Table I tell that applying CorrReg, with any number nReg of two-way partitions, to these layers consistently achieves performance boost over the LeNet variant baseline. In general, CorrReg is more effective for (upper) FC layers; this is reasonable since a densely connected FC layer contains much more trainable parameters than a conv layer does, and is thus more susceptible to overfitting. Setting $\texttt{nReg} > 1$ sometimes helps in getting even better results, but at the cost of slightly increased computation. In the subsequent experiments, we simply set $\texttt{nReg} = 1$ for computational efficiency.

As a technique for regularization of network training, CorrReg is related to the methods [16], [18], [35] discussed in Section IV, in particular Dropout [18]. To compare CorrReg with Dropout, we apply them to the FC5 layer of LeNet variant. Since their working mechanisms are different, one might be also interested in using them together. Table II reports the comparative results. CorrReg achieves improvement comparable to that of Dropout, where the dropping rate is optimally set as 0.2 by tuning from the range of $(0, 1)$ on the validation set. Using CorrReg together with Dropout further improves the performance, showing the complementary regularization benefit of CorrReg to that of Dropout.

TABLE II

COMPARISON OF IMAGE CLASSIFICATION ON CIFAR-10 [23] WHEN APPLYING CORRREG AND/OR DROPOUT TO AN UPPER FC LAYER OF A VARIANT OF LENET [24]. EXPERIMENTS ARE RUN FOR 5 TIMES, AND RESULTS ARE IN THE FORMAT OF MEAN (STANDARD DEVIATION)

Methods	Error rates (%)
Plain LeNet variant	17.42 (0.16)
Dropout [18]	16.72 (0.18)
CorrReg	16.75 (0.20)
CorrReg + Dropout	16.36 (0.25)

TABLE III

EXPERIMENTS ON CIFAR-10 [23] USING A VARIANT OF LENET [24]. CORRREG IS OPTIONALLY APPLIED TO AN UPPER FC LAYER WITH VARYING NUMBERS OF LAYER NEURONS. RESULTS ARE IN THE FORMAT OF ERROR RATE (%)/CORRELATION COEFFICIENT ($1e^{-2}$). REFER TO THE MAIN TEXT FOR HOW CORRELATION IS COMPUTED

Neuron No.	128	256	512	1024
W/O CorrReg	17.85/0.53	17.42/0.70	17.17/0.76	17.01/0.79
With CorrReg	17.35/0.65	16.75/0.75	16.35/0.79	16.08/0.82

CorrReg achieves regularization via improving correlations between the internal features produced by two-way partition of a layer, which suggests a natural alternative that halves the number of layer neurons. Halving the number of layer neurons reduces the model capacity and creates "bottlenecking" of information flow, thus implicitly imposing regularization. Oppositely, one may be also interested in alternatives that increase the number of layer neurons with varying factors. To investigate the efficacy of CorrReg for models with different capacities, we apply these alternatives again to the FC5 layer of LeNet variant. Results in Table III show that larger models perform better than smaller ones do, and applying our proposed CorrReg to larger models further improves the performance. We also compute in Table III the averaged (pair-wise) correlation among features of training samples learned at different layer neurons, in order to understand how network capacities relate to the behavior of feature correlations across layer neurons and how CorrReg plays a role here as a regularization. Results show that as the numbers of layer neurons increase, feature correlations between layer neurons increase, and applying CorrReg further enhances this effect.

CorrReg is a neuron-wise scheme of CCA regularization. One might be interested in the performance when using CCA as the regularizer. To this end, we apply the CCA objective (2) as a regularizer to the FC5 layer of the LeNet variant, where regularization parameter is set as $1e^{-6}$ by optimally tuning on the validation set. Computational complexity of CCA regularization is significantly worse than that of CorrReg, and Table IV shows that it practically consumes more time per iteration of SGD training (measured on an M40 GPU and Intel Xeon CPU running at 2.2 GHz). Although CCA regularization improves performance over that of plain LeNet variant, its results with different sizes of mini-batches are worse than those of CorrReg; we hypothesize that this is because optimization of CCA objective (particularly the constraints in (2)) is incompatible with SGD based network training.

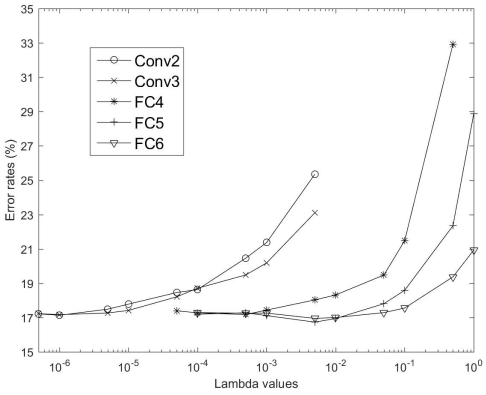

Fig. 4. Investigation of the penalty parameter λ on CorrReg's performance. Each line represents the validation errors on CIFAR-10 [23] when applying CorrReg to a layer of a variant of LeNet [24] using a range of λ values. Each error rate is a mean out of 5 runs. These results are obtained by using 10% of CIFAR-10 training samples as validation.

The penalty parameter λ in (4) controls the amount of regularization that CorrReg imposes on network training. To investigate how performance of CorrReg depends on λ values, we use 10% of training samples as validation, and apply CorrReg to different layers of the LeNet variant (i.e., the settings of Table I with $\texttt{nReg} = 1$) using a range of λ values. Results are plotted in Figure 4. Figure 4 shows that smaller values of λ are usually better when applying CorrReg to (lower) conv layers, and larger values of λ are usually better for (upper) FC layers. This is reasonable due to two compound reasons: (1) when applying CorrReg to an upper network layer, larger values of λ are needed in order to back-propagate the regularization for better learning of features/parameters of all the layers below; (2) conv layers already have intrinsic regularization via weight sharing. This inconsistency of optimal λ values across different network layers makes use of CorrReg less convenient. Fortunately, results in this section suggest that to get the most effective regularization, one may simply apply CorrReg to an upper (FC) network layer, and set the optimal λ values accordingly. Setting $\lambda \in [1e^{-3}, 1e^{-1}]$ typically gives good results. Experiments in the subsequent sections follow this empirical rule.

1) Results on Modern Deep Architectures: We further investigate the regularization effects of CorrReg on modern deep architectures. For the datasets of CIFAR-10 and CIFAR-100, we use the representative architectures of ResNet [63], Wide ResNet [26], and DenseNet [27]. The CIFAR-100 dataset is an adaptation of CIFAR-10, consisting of 100 object categories of $60,000$ color images. We use simple data augmentation following [62]: during training, we zero-pad 4 pixels along each image side, and sample a 32×32 region crop from the padded image or its horizontal flip; during testing, we simply use the original non-padded image. Our use of ResNet, Wide ResNet, and DenseNet for the CIFAR datasets is as follows: we use a pre-activation ResNet [63] of 68 weight layers, whose layer specifics are given in supplementary material; we use the exactly same top-performing architecture of "WRN-28-10" as in [26]; we also use the exactly same top-performing architecture of "DenseNet-BC" (with the growth

TABLE IV

COMPUTATION AND RECOGNITION PERFORMANCE ON CIFAR-10 [23] WHEN APPLYING CORRREG OR CCA REGULARIZATION, WITH DIFFERENT SIZES m OF MINI-BATCHES, TO AN UPPER FC LAYER OF A VARIANT OF LENET [24]. WALL-CLOCK TIME IS MEASURED ON AN M40 GPU AND INTEL XEON CPU RUNNING AT 2.2 GHZ. EXPERIMENTS ARE RUN FOR 5 TIMES, AND ACCURACIES ARE IN THE FORMAT OF MEAN (STANDARD DEVIATION). $n_l = 256$ AND $n_{l+1} = 64$ DENOTE THE NUMBERS OF INPUT AND OUTPUT NEURONS OF THE LAYER RESPECTIVELY

Methods	Plain LeNet variant			CorrReg			CCA regularization		
	$m = 128$	$m = 256$	$m = 512$	$m = 128$	$m = 256$	$m = 512$	$m = 128$	$m = 256$	$m = 512$
Computational Complexity	$\mathcal{O}(mn_l n_{l+1})$			$\mathcal{O}(mn_l n_{l+1})$			$\mathcal{O}(mn_l n_{l+1} + mn_{l+1}^2 + n_{l+1}^3)$		
wall-clock time per iter. (sec.)	0.009	0.016	0.030	0.013	0.020	0.035	0.025	0.035	0.048
Error rates (%)	17.42 (0.16)	17.64 (0.17)	18.33 (0.19)	16.75 (0.20)	17.13 (0.18)	17.43 (0.12)	17.24 (0.33)	17.39 (0.22)	17.65 (0.24)

TABLE V

RESULTS (ERROR RATES %) ON CIFAR-10 AND CIFAR-100 [23] OF SEVERAL DEEP ARCHITECTURES WITH OR WITHOUT REGULARIZATION. STANDARD DATA AUGMENTATION IS USED AS IN [62]. REFER TO THE MAIN TEXT FOR HOW DROPOUT-S1, DROPOUT-S1S2, CORRREG, AND CORRREG-DROPOUTS2 ARE APPLIED TO THESE ARCHITECTURES

Architecture	CIFAR-10					CIFAR-100				
	W/O Regu.	Dropout-S1	Dropout-S1S2	CorrReg	CorrReg-DropoutS2	W/O Regu.	Dropout-S1	Dropout-S1S2	CorrReg	CorrReg-DropoutS2
ResNet [63]	6.69	6.25	6.13	6.15	**6.02**	27.68	28.08	26.58	27.32	**26.33**
Wide ResNet [26]	4.39	4.50	4.22	4.11	**4.05**	21.40	22.07	20.75	21.32	**20.38**
DenseNet [27]	3.90	3.81	3.75	**3.45**	3.51	18.99	18.19	18.15	18.06	**17.99**

rate $k = 40$) as in [27]. These architectures commonly aggregate features of lower layers via a top global average pooling layer, followed by a final FC layer of classification. For each network, we follow the empirical rule established in Section V-A and apply CorrReg to the final FC layer. We fix λ of CorrReg as $5e^{-3}$ for all the three networks. We train ResNet and Wide ResNet for a total of 160 epochs; learning rates are initialized as 0.1, and decay after 80 and 120 epoches of training. For DenseNet, we follow [27] and train for an extended duration of 300 epochs, using mini-batches of size 64. All other training hyperparameters are the same as described in the beginning of Section V (not necessarily the same as used in [26], [27], [63]). To compare with Dropout regularization, we use two schemes (denoted as Dropout-S1 and Dropout-S1S2 respectively): scheme 1 applies Dropout to (inputs of) the final FC layer of each network, which is the same as our use of CorrReg; scheme 2 follows the way in [26] and *additionally* applies Dropout to (inputs of) the second one of the two conv layers in each residual block of these networks. Dropping rates are tuned on the validation set with the optimal one set as 0.2. We also try CorrReg together with the above scheme 2 (denoted as CorrReg-DropoutS2), to compare fairly with Dropout regularization. Results in Table V show that Dropout-S1S2 improves over Dropout-S1 by providing additional regularization, especially for the CIFAR-100 dataset that contains much fewer training samples per category than CIFAR-10 does. When applying to the top FC layer alone, our proposed CorrReg consistently outperforms Dropout-S1. Moreover, the best results are obtained by CorrReg-DropoutS2 that has the combined benefit of CorrReg and Dropout regularization.

For the ImageNet dataset, we use the representative architectures of ResNet [25], Wide ResNet [26], and ResNeXt [57] (more specifically, the "ResNet-101", and the top-performing "WRN-50-2-bottleneck" and "ResNeXt-101 (64×4d)" of these architectures). For data augmentation, we adopt the same scheme as in [25]: during training, we randomly sample a 224×224 region crop from an image or its horizontal flip; during testing, we use a single crop of size 224×224. Learning rates are initialized as 0.1, and decay by a factor of 0.1 at 50% and 75% of the total 90 training epochs, using mini-batches of size 256. For each network, we again apply CorrReg to the top FC layer, using a fixed λ value of $1e^{-3}$. We also apply Dropout to (inputs of) the same FC layers of these networks, where dropping rate is again set as 0.2. Table VI shows the comparative results. While Dropout regularization may not have effect on these architectures, our proposed CorrReg steadily achieves performance improvement.

TABLE VI

RESULTS OF SINGLE-CROP TESTING ON THE IMAGENET VALIDATION SET [58] OF SEVERAL DEEP ARCHITECTURES WITH OR WITHOUT REGULARIZATION. RESULTS ARE IN THE FORMAT OF TOP-1/TOP-5 ERROR RATES (%)

Architecture	W/O Regu.	With Dropout	With CorrReg
ResNet [25]	22.39/6.25	22.59/6.30	**22.14/6.09**
Wide ResNet [26]	22.22/6.35	22.25/6.25	**22.10/6.17**
ResNeXt [57]	20.90/5.46	20.82/5.70	**20.58/5.40**

Remarks We note that experiments in this section are not intended to compare with the best results on the benchmark image classification datasets. They are to show the efficacy of our proposed CorrReg for regularization of network training: even though input data are from the same source, intermediate features may be learned to be overfitting to view-specific patterns. CorrReg effectively regularizes network training so that the final task of classification can collaboratively benefit from all feature views.

B. RGB-D Object and Scene Recognition

In this section, we use RGB-D object dataset [28] and SUN RGB-D scene dataset [29] to investigate the efficacy of

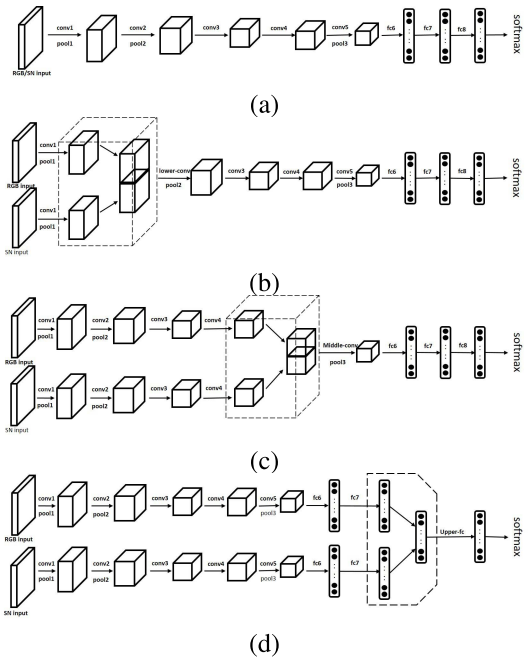

Fig. 5. Network architectures used for experiments of RGB-D object recognition. The corresponding layer parameters (numbers of feature maps, filter sizes, etc) are given in supplementary material. (a) The plain (RGB or depth) ConvNets. (b)-(d) The RGB-Depth ConvNets concatenating features of the two lower streams respectively at lower, middle, or upper "heights" of network layers.

TABLE VII

RECOGNITION ACCURACIES(%) ON RGB-D OBJECT DATASET [28] USING ARCHITECTURES IN FIGURE 5 AND VARIOUS REGULARIZATION METHODS. RESULTS ARE IN THE FORMAT OF MEAN ± STANDARD DEVIATION

Methods	Accuracy
RGB ConvNet	79.83 ± 2.06
Depth Convnet	83.49 ± 2.00
RGB-Depth ConvNet (concat. at a lower height)	79.48 ± 2.64
RGB-Depth ConvNet (concat. at a middle height)	79.56 ± 3.66
RGB-Depth ConvNet (concat. at an upper height)	87.99 ± 1.51
RGB-Depth ConvNet with Dropout	88.33 ± 1.69
RGB-Depth ConvNet with L2Regu	88.65 ± 1.37
RGB-Depth ConvNet with CorrReg	$\mathbf{89.16 \pm 1.18}$
RGB-Depth ConvNet with L2Regu & Dropout	88.58 ± 1.85
RGB-Depth ConvNet with CorrReg & Dropout	$\mathbf{89.65 \pm 1.40}$

TABLE VIII

ACCURACIES(%) OF DIFFERENT λ VALUES WHEN APPLYING CORRREG TO A RGB-DEPTH CONVNET (FIGURE 5-(D)) FOR RGB-D OBJECT RECOGNITION [28]

λ	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
mean	88.18	87.86	88.08	89.16	89.15	86.08
±std	±1.22	±1.85	±1.26	±1.18	±1.93	±1.39

CorrReg for practical problems of multi-view learning. The RGB-D object dataset contains 207, 920 RGB-D video frames of 51 classes of 300 object instances captured from different views, with roughly 600 frames per object instance. We sample these videos by every 5^{th} frame of each video. We use the 10 random dataset splits provided by [28], with each split containing different object instances of all the 51 classes. For these splits, there are on average about 34, 000 images for training and 6, 900 images for testing. This dataset is intensively used for comparative studies of alternative baselines, investigation of our proposed method under different settings, and also for robustness test against contamination of input data. SUN RGB-D scene dataset is a benchmark suite for indoor scene understanding, including 10, 355 RGB-D images. For scene recognition, we follow [29] and select the 19 most common categories, each of which has at least 80 RGB-D images in the dataset; we then divide the data into training and test sets, giving 4, 845 images for training and 4, 659 ones for testing. For both RGB-D object and SUN RGB-D datasets, we compute surface normal (SN) [64] from each depth image as input depth features.

1) Comparative Studies of Alternative Baselines: Our comparative studies of alternative baselines for RGB-D object recognition are based on adaptations of an 8-layer ConvNet. The adaptations consist of two lower, parallel streams followed by one upper, single stream, whose model architectures are shown in Figure 5 and whose layer parameters (numbers of feature maps, filter sizes, etc) are given in supplementary material. RGB and depth/SN images are respectively taken as inputs of the two lower streams, whose outputs are concatenated as inputs of the upper stream. We term such adapted networks as RGB-Depth ConvNets. In this work, we investigate the effects of concatenating features of the two lower streams at different "heights" (lower, middle, or upper layers) of RGB-Depth ConvNets, as illustrated in Figure 5. Alternative to RGB-Depth ConvNets are plain ConvNets that consist of one of the two lower streams of RGB-Depth ConvNets and the upper stream, as shown in Figure 5. Such plain ConvNets can be used for both RGB and depth images, and we term them as RGB ConvNet and Depth ConvNet respectively.

The above networks suggest several baseline methods for RGB-D object recognition. In particular, given existence of both RGB and depth images during training and test phases, one may separately train RGB ConvNet or Depth ConvNet using single-modal images, and use the trained models for single-modal inference. Alternatively, one may use the above RGB-Depth ConvNets that concatenate features of individual modalities at different heights for multi-modal training and inference. To train these baseline networks, we use common data augmentation practices on the RGB-D object dataset: we first re-scale each training image to the size of 150 × 150, from which or the horizontal flip of which we randomly crop a region of the size 143 × 143. The learning rate is initialized as 0.01 and decays by a factor of 0.1 when learning curves plateau. Table VII shows that while concatenating features at lower or middle layers of RGB-Depth ConvNets is not effective, feature concatenation at an upper layer of RGB-Depth ConvNet achieves improved performance over single-modal networks.

The above baselines fuse multi-modal features via direct concatenation. Discussions in Section I suggest that one may apply *network regularization at fusion* to help collaboratively learn from multi-view features, which includes existing meth-

TABLE IX

ROBUSTNESS TEST BY ADDING RANDOM OCCLUSION BLOCKS OF VARYING SIZES TO TEST RGB AND DEPTH (SN) IMAGES OF THE RGB-D OBJECT DATASET [28]. TRAINED NETWORKS IN SECTION V-B.1 ARE USED FOR THESE EXPERIMENTS. RESULTS ARE IN TERMS OF RECOGNITION ACCURACY(%)

Occlusion size	10×10	20×20	30×30	40×40	50×50
RGB ConvNet	79.25	75.29	63.52	45.58	28.35
Depth ConvNet	83.89	82.71	80.07	71.96	54.67
RGB-Depth ConvNet	88.05	86.81	80.60	66.32	43.53
RGB-Depth ConvNet with Dropout	88.34	86.98	80.71	66.10	44.66
RGB-Depth ConvNet with CorrReg	89.27	88.66	**85.33**	**75.73**	**56.75**
RGB-Depth ConvNet with CorrReg & Dropout	**89.70**	**88.93**	84.54	73.28	53.46

ods such as Dropout [18], and also our proposed CorrReg that explicitly leverages multi-view learning criteria. More specifically, we apply CorrReg to the first layer of the upper stream of RGB-Depth ConvNet that concatenates multi-view features at upper "height", making it become a CorrReg fusion layer, or alternatively apply Dropout to inputs of the first layer of the upper stream. As noted in Section IV, an approximate measure of correlation is used in [47] that simply computes the (squared) Euclidean distance between features of individual views. We also consider this simple correlation measure as a baseline regularizer of (1), and term such a method as L2Regu. We compare with L2Regu by applying it to the same first layer of the upper stream of RGB-Depth ConvNet. We set the λ value of CorrReg as $1e^{-1}$, the same penalty parameter of L2Regu as $1e^{-1}$, and the dropping rate of Dropout as 0.5, which are determined by tuning on the validation set. Results in Table VII show that either Dropout, L2Regu, or CorrReg improves performance over that of direct concatenation, and CorrReg outperforms Dropout and L2Regu, showing the advantage of CorrReg in practical multi-view learning problems. Table VIII also gives results of CorrReg when using different λ values. To investigate whether the effect of CorrReg (or L2Regu) is complementary to that of Dropout, we also use both CorrReg (or L2Regu) and Dropout in RGB-Depth ConvNet. Using L2Regu together with Dropout may not improve over L2Regu itself; instead, using CorrReg together with Dropout further improves the performance, showing the advantage of CorrReg for complementary regularization.

2) Robustness against Contamination of Input Data: An important property of multi-view learning is that inference should be less influenced when input data are contaminated [65]. In this section, we simulate such testing scenarios by adding random occlusion blocks to input RGB and depth images. Occlusion blocks are obtained by setting pixel values of the occluded regions as 0. We use the trained networks in Section V-B.1 for these investigations. Table IX reports comparative results under different sizes of random occlusion. Compared with RGB ConvNet and Depth ConvNet, direct feature concatenation using RGB-Depth ConvNet may not provide better robustness against contamination of input data. RGB-Depth ConvNet with our proposed CorrReg improves the robustness, and performs consistently better than plain RGB-Depth ConvNet and the one with Dropout do.

3) Comparisons with the state of the art: State-of-the-art results on RGB-D object recognition are obtained by using either advanced base models (e.g., ResNets [25]) with parameters pre-trained on ImageNet [47], or advanced feature encoding scheme [48]. We follow [47] and use two ResNet-50 [25] (after removing its last FC layer of classifier) as the lower, parallel streams, whose 2048-dimensional output feature vectors are concatenated as the input vector of a FC based CorrReg fusion layer, followed by the last layer of 51-way classifier. The two lower streams respectively take RGB and SN images as inputs. We term such a constructed network as RGB-Depth ResNet. For data augmentation, we follow [46] by first re-scaling training images to the size of 256×256, and then randomly cropping regions of the size 224×224 from them or their horizontal flips. RGB-Depth ResNet is fine-tuned with the initial learning rates of 0.0001 for the lower streams and 0.01 for the upper stream. We use mini-batches of size 64, and set the λ value of CorrReg as $1e^{-1}$ and the dropping rate of Dropout as 0.8. Other training hyperparameters are the same as described in the beginning of Section V.

TABLE X

RECOGNITION ACCURACIES (%) OF DIFFERENT METHODS ON THE RGB-D OBJECT DATASET [28]. RESULTS ARE IN THE FORMAT OF MEAN $\pm$ STANDARD DEVIATION

Methods	Accuracy
Nonlinear SVM[28]	83.9 ± 3.5
CKM [66]	86.4 ± 2.3
CNN-RNN [14]	86.8 ± 3.3
upgraded HMP [64]	87.5 ± 2.9
MMSS [46]	88.5 ± 2.2
Fus-CNN [45]	91.3 ± 1.4
CIMDL-ResNet [47]	92.4 ± 1.8
MDSI-CNN [48]	92.8 ± 1.2
RGB ResNet	90.5 ± 1.6
Depth ResNet	85.5 ± 2.4
RGB-Depth ResNet	92.5 ± 1.2
RGB-Depth ResNet with Dropout	93.1 ± 1.4
RGB-Depth ResNet with CorrReg	$\mathbf{93.4 \pm 1.6}$
RGB-Depth ResNet with CorrReg & Dropout	$\mathbf{93.6 \pm 1.6}$

Note that the method [47] uses the same ImageNet pre-trained base models (i.e., ResNet-50) as we do. It is interesting to observe in Table X that our result of RGB-Depth ResNet that concatenates RGB and depth features directly is better than those from most of existing methods. Regularizing RGB-Depth ResNet with either Dropout or CorrReg further improves the result, with CorrReg achieving better improvement. Using CorrReg together with Dropout achieves the new state of the art of 93.6%.

TABLE XI

RECOGNITION ACCURACIES (%) OF DIFFERENT METHODS ON THE SUN RGB-D SCENE DATASET [29]

Methods	Accuracy
GIST + RBF Kernel SVM [29]	23.0
Place-CNN + Linear SVM [29]	37.2
Place-CNN + RBF Kernel SVM [29]	39.0
SSCNN [50]	41.3
DMFF [67]	41.5
MDSI-CNN [48]	45.2
FV-CNN [51]	48.1
RGB-D-CNN(wSVM) [52]	52.4
BilinearCNN [53]	55.5
RGB ResNet	57.8
Depth ResNet	47.2
RGB-Depth ResNet	59.6
RGB-Depth ResNet with Dropout	60.0
RGB-Depth ResNet with CorrReg	**60.7**
RGB-Depth ResNet with CorrReg & Dropout	**61.0**

4) Results on RGB-D Scene Recognition: In this section, we report experiments of RGB-D scene recognition on the SUN RGB-D dataset [29]. We use the same network architectures and training manners as in Section V-B.3, with the only difference that replaces the 51-way softmax classifiers with the 19-way ones. Results in Table XI tell that RGB-Depth ResNet using simple feature concatenation outperforms existing methods that have complicated feature fusion schemes and/or training criteria. Regularizing RGB-Depth ResNet with CorrReg and/or Dropout further improves the result to the new state of the art.

C. Multi-view Recognition of 3D Object Shapes

We conduct experiments of multi-view 3D object recognition on the ModelNet40 dataset [59] to investigate the efficacy of CorrReg for practical problems with data of more than two views. The ModelNet40 dataset contains 12,311 CAD models (meshes) of 40 object categories, with 9,843 models for training and 2,468 ones for testing. To prepare images of multiple views from each object model, we follow the 1^{st} camera set-up in [54] and assume that each model is upright oriented; 12 virtual cameras, pointing towards the model centroid, are evenly distributed (with intervals of 30 degrees) around a horizontal circle that is elevated 30 degrees from the ground plane; 2D images are rendered from these 12 camera views.

Based on a very simple architecture of MVCNN [54] for multi-view based 3D object recognition, where features of individual views extracted from lower, parallel layer streams are aggregated via feature-wise max pooling, we design a Multi-view Fusion Network (MvFusionNet) as shown in Figure 6. By pairing neighboring views, MvFusionNet re-organizes feature vectors of individual views from lower streams into an equal number of pairs of feature vectors. Each of such pairs is then fed into a fusion layer where CorrReg can also be applied to form a CorrReg fusion layer. Feature-wise max pooling is subsequently applied to outputs of these fusion layers, and MvFusionNet ends with a FC layer of classifier. We investigate here whether CorrReg is helpful for feature aggregation of different views by regularizing such a constructed MvFusionNet.

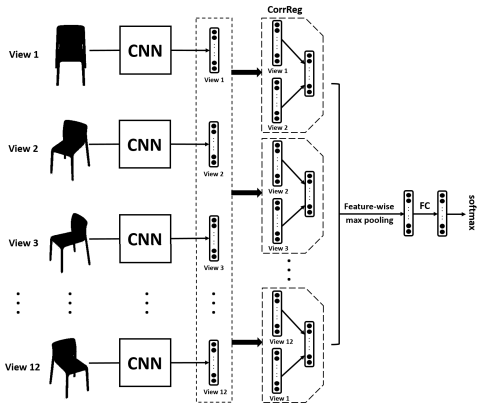

Fig. 6. Illustration of the multi-view fusion network used for experiments of multi-view 3D object recognition. Thick right arrows represent data re-organization by pairing feature vectors of individual views.

TABLE XII

ACCURACIES (%) OF MULTI-VIEW 3D OBJECT RECOGNITION ON THE MODELNET40 DATASET [59]. ALL METHODS USE THE 1^{st} CAMERA SET-UP IN [54] (12 CAMERA VIEWS POINTING TOWARDS UPRIGHT ORIENTED MODEL). BEST RESULTS OF SOME METHODS AND THE CORRESPONDING VIEW NUMBERS ARE ALSO QUOTED IN PARENTHESES

Methods	Accuracy
MVCNN [54]	89.9 (90.1, 80 views)
Pairwise [70]	90.7
Dominant Set Clustering [68]	92.2 (93.0, 24 views)
MHBN [55]	93.4 (94.1, 6 views)
GVCNN [56]	92.6 (93.1, 8 views)
MvFusionNet	93.9
MvFusionNet with CorrReg	**95.4**

Lower streams of MvFusionNet are adapted from ResNet-101 [25] that is pre-trained on ImageNet [47]. To train MvFusionNet, we use a mini-batch of 16 (i.e., $16 \times 12 = 192$ images); the learning rates start at 0.001 and decay at the rate of 0.1 when learning curves plateau. The penalty λ of CorrReg is set as $5e^{-4}$. We report in Table XII results of MvFusionNet without or with CorrReg regularization, where we also compare with recent state-of-the-art results [55], [56], [68] on ModelNet40 whose multi-view images are prepared following the same style of 1^{st} camera set-up in [54] (i.e., 12 camera views pointing towards upright orientation of object models). Due to varying architectural designs, network optimizers, and feature aggregation schemes, results of different methods in Table XII may not be directly comparable; nevertheless, it confirms the efficacy of CorrReg for better feature learning and aggregation from multiple views of 3D object shapes. We note that results of multi-view based methods on ModelNet40 depend heavily on how multi-view images are prepared by positioning virtual cameras on a sphere enclosing the object model. For example, the current best result on ModelNet40 is obtained in [69] by selecting camera set-ups from a much richer set of camera positioning and viewpoints. We expect our results can also be boosted by using multi-view images rendered from these optimal camera set-ups.

VI. Conclusion

We study in this paper deep multi-view learning in the context of regularized network training. We take a regularization approach via multi-view learning criteria, and propose a novel, effective, and efficient neuron-wise correlation-maximizing regularizer. We also implement such regularizers collectively as a correlation-regularized network layer (CorrReg). CorrReg can be applied to either FC or conv based fusion layers that concatenate intermediate features of individual views. Controlled experiments of benchmark image classification show that CorrReg consistently improves performance of various modern deep architectures. Applying CorrReg to multi-modal deep networks achieves the new state of the art on the benchmark RGB-D object and scene recognition datasets. In future research, we are interested in applying CorrReg to other multi-view learning problems of practical interest.

References

[1] H. Hotelling, "'Relations between two sets of variates," *Biometrika*, vol. 28, nos. 3–4, pp. 321–377, Dec. 1936.

[2] G. Andrew, R. Arora, K. Livescu, and J. Bilmes, "Deep canonical correlation analysis," in *Proc. Int. Conf. Mach. Learn.*, Feb. 2013, pp. 1247–1255.

[3] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2011, pp. 689–696.

[4] A. Frome *et al.*, "Devise: A deep visual-semantic embedding model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2121–2129.

[5] F. R. Bach and M. I. Jordan, "A probabilistic interpretation of canonical correlation analysis," Dept. Statist., Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. 688, 2005.

[6] D. A. Cohn and T. Hofmann, "The missing link - a probabilistic model of document content and hypertext connectivity," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 430–436.

[7] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan, "Matching words and pictures," *J. Mach. Learn. Res.*, vol. 3, pp. 1107–1135, Feb. 2003.

[8] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 2222–2230.

[9] R. Arora, A. Cotter, K. Livescu, and N. Srebro, "Stochastic optimization for PCA and PLS," in *Proc. 50th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Monticello, IL, USA, Oct. 2012, pp. 861–868.

[10] D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Comput.*, vol. 16, no. 12, pp. 2639–2664, Dec. 2004.

[11] P. L. Lai and C. Fyfe, "Kernel and nonlinear canonical correlation analysis," *Int. J. Neural Syst.*, vol. 10, pp. 365–377, Oct. 2000.

[12] D. R. Hardoon and J. Shawe-Taylor, "Sparse canonical correlation analysis," *Mach. Learn.*, vol. 83, no. 3, pp. 331–353, 2011.

[13] W. Wang, R. Arora, K. Livescu, and J. Bilmes, "On deep multi-view representation learning," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2015, pp. 1083–1092.

[14] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng, "Convolutional-recursive deep learning for 3D object classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 656–664.

[15] S. Akaho, "A kernel method for canonical correlation analysis," in *Proc. Int. Meeting Psychometric Soc.*, May 2016, pp. 1–15.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[17] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. 13th Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 818–833.

[18] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, pp. 1–18, Jul. 2012.

[19] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. Cambridge, MA, USA, MIT Press, 2012.

[20] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, Feb. 2017, pp. 1–15.

[21] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1225–1234.

[22] I. Kuzborskij and C. H. Lampert, "Data-dependent stability of stochastic gradient descent," *CoRR*, vol. abs/1703.01678, pp. 1–22, Mar. 2017.

[23] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[26] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, May 2016, pp. 1–15.

[27] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, pp. 1–9, Aug. 2016.

[28] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 1817–1824.

[29] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 567–576.

[30] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. 30th Int. Conf. Mach. Learn.*, May 2013, pp. 1139–1147.

[31] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[32] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *CoRR*, vol. abs/1602.07868, pp. 1–11, Feb. 2016.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, pp. 1–14, Sep. 2014.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[35] L. Wan, M. D. Zeiler, S. Zhang, Y. LeCun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proc. Int. Conf. Mach. Learn.*, Feb. 2013, pp. 1058–1066.

[36] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Representations*, Dec. 2013, pp. 1–10.

[37] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, "Striving for simplicity: The all convolutional net," *CoRR*, vol. abs/1412.6806, pp. 1–14, Dec. 2014.

[38] P. Baldi and P. J. Sadowski, "Understanding dropout," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2814–2822.

[39] S. Wager, S. Wang, and P. S. Liang, "Dropout training as adaptive regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 351–359.

[40] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," in *Proc. Int. Conf. Learn. Representations*, Jun. 2016, pp. 1–12.

[41] P. Rodríguez, J. Gonzàlez, G. Cucurull, J. M. Gonfaus, and X. Roca, "Regularizing cnns with locally constrained decorrelations," in *Proc. Int. Conf. Learn. Representations*, Mar. 2017, pp. 1–11.

[42] K. Jia, D. Tao, S. Gao, and X. Xu, "Improving training of deep neural networks via singular value bounding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3994–4002.

[43] S. Chandar, M. M. Khapra, H. Larochelle, and B. Ravindran, "Correlational neural networks," *CoRR*, vol. abs/1504.07225, pp. 1–27, Apr. 2015.

[44] L. Bo, X. Ren, and D. Fox, "Hierarchical matching pursuit for image classification: Architecture and fast algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 2115–2123.

[45] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep./Oct. 2015, pp. 681–687.

[46] A. Wang, J. Cai, J. Lu, and T.-J. Cham, "MMSS: Multi-modal sharable and specific feature learning for RGB-D object recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1125–1133.
[47] Z. Wang, J. Lu, R. Lin, J. Feng, and J. Zhou, "Correlated and individual multi-modal deep learning for RGB-D object recognition," *CoRR*, vol. arXiv:1604.01655, pp. 4321–4330, Apr. 2016.
[48] U. Asif, M. Bennamoun, and F. A. Sohel, "A multi-modal, discriminative and spatially invariant CNN for RGB-D object labeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 9, pp. 2051–2065, Sep. 2018.
[49] X. Li, M. Fang, J.-J. Zhang, and J. Wu, "Learning coupled classifiers with RGB images for RGB-D object recognition," *Pattern Recognit.*, vol. 61, pp. 433–446, Jan. 2017.
[50] Y. Liao, S. Kodagoda, Y. Wang, L. Shi, and Y. Liu, "Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 2318–2325.
[51] A. Wang, J. Cai, J. Lu, and T.-J. Cham, "Modality and component aware feature fusion for RGB-D scene classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5995–6004.
[52] X. Song, L. Herranz, and S. Jiang, "Depth CNNs for RGB-D scene recognition: Learning from scratch better than transferring from RGB-CNNs," in *Proc. AAAI*, Feb. 2017, pp. 4271–4277.
[53] H. F. M. Zaki, F. Shafait, and A. Mian, "Learning a deeply supervised multi-modal RGB-D embedding for semantic scene and object category recognition," *Robot. Auto. Syst.*, vol. 92, pp. 41–52, Jun. 2017.
[54] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks For 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 945–953.
[55] T. Yu, J. Meng, and J. Yuan, "Multi-view harmonized bilinear network for 3D object recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 186–194.
[56] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao, "GVCNN: Group-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 264–272.
[57] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5987–5995.
[58] O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[59] Z. Wu *et al.*, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. CVPR*, Jun. 2015, pp. 1912–1920.
[60] R. Collobert, S. Bengio, and J. Mariethoz, "Torch: A modular machine learning software library," IDIAP, Res. Rep. 02–46, 2002.
[61] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout networks," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 1319–1327.
[62] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. 8th Int. Conf. Artif. Intell. Statist.*, Feb. 2015, pp. 562–570.
[63] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.
[64] L. Bo, X. Ren, and D. Fox, "Unsupervised feature learning for RGB-D based object recognition," in *Experimental Robotics*. Cham, Switzerland: Springer, 2013, pp. 387–402.
[65] C. Xu, D. Tao, and C. Xu, "Multi-view intact space learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 12, pp. 2531–2544, Dec. 2015.
[66] M. Blum, J. T. Springenberg, J. Wülfing, and M. Riedmiller, "A learned feature descriptor for object recognition in RGB-D data," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 1298–1303.
[67] H. Zhu, J.-B. Weibel, and S. Lu, "Discriminative multi-modal feature fusion for rgbd indoor scene recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2969–2976.
[68] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling formulti-view 3D object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2017, pp. 1–12.
[69] A. Kanezaki, Y. Matsushita, and Y. Nishida, "Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5010–5019.
[70] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proc. CVPR*, Jun. 2016, pp. 3813–3822.

Kui Jia received the B.Eng. degree in marine engineering from Northwestern Polytechnical University, China, in 2001, the M.Eng. degree in electrical and computer engineering from the National University of Singapore in 2003, and the Ph.D. degree in computer science from the Queen Mary University of London, London, U.K., in 2007. He is currently a Professor with the School of Electronic and Information Engineering, South China University of Technology. His research interests include computer vision, machine learning, and image processing. In these areas, he has authored many publications in prestigious journals and conferences, such as the IEEE T-PAMI, IJCV, T-IP, T-SP, CVPR, ICCV, and ECCV. His recent research focuses on theoretical deep learning and its applications in semantic analysis and reconstruction of data beyond the Euclidean domain.

Jiehong Lin received the bachelor's degree from the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China, in 2017, where he is currently pursuing the master's/Ph.D. degrees in information and communication engineering. His research interests include 3D vision and machine learning.

Mingkui Tan received the bachelor's degree in environmental science and engineering and the master's degree in control science and engineering from Hunan University, Changsha, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from Nanyang Technological University, Singapore, in 2014. From 2014 to 2016, he was a Senior Research Associate of computer vision with the School of Computer Science, University of Adelaide, Australia. He is currently a Professor with the School of Software Engineering, South China University of Technology. His research interests include machine learning, sparse analysis, deep learning, and large-scale optimization.

Dacheng Tao (F'15) is currently a Professor of computer science and an ARC Laureate Fellow with the School of Computer Science and the Faculty of Engineering and Information Technologies, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Centre, The University of Sydney. He mainly applies statistics and mathematics to artificial intelligence and data science. His research results have expounded in one monograph and more than 200 publications at prestigious journals and prominent conferences, such as IEEE T-PAMI, T-IP, T-NNLS,T-CYB, IJCV, JMLR, NIPS, ICML, CVPR, ICCV, ECCV, ICDM, and ACM SIGKDD, with several best paper awards, such as the Best Theory/Algorithm Paper Runner Up Award in IEEE ICDM07, the Best Student Paper Award in IEEE ICDM13, the 2014 ICDM 10-year Highest-Impact Paper Award, the 2017 IEEE Signal Processing Society Best Paper Award, and the Distinguished Paper Award in the 2018 IJCAI. He received the 2015 Austrlian Scopus-Eureka Prize and the 2018 IEEE ICDM Research Contributions Award. He is a fellow of the Australian Academy of Science, AAAS, IAPR, OSA, and SPIE.